REMARKS/ARGUMENTS

Amendments were made to the specification. No new matter has been added by any of the amendments to the specification.

Claims 1-34 are pending in the present application. Claims 1, 2, 8-10, 12, 13, 17, 19-21, 23, 24, 30-32, and 34 are amended. Support for the claim amendments can be found in original, now canceled, claims 7, 18 and 29, and on page 26 of the specification. Claims 7, 18, and 29 are canceled. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 112, Second Paragraph

The Examiner has noted the use of the Trademark JAVA®. Applicant has amended the specification to address the Examiner's concerns.

The Examiner has rejected claims 1-34 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter, which Applicants regard as the invention. Specifically, the Examiner objects to the use of the Trademark "JAVA" within the claim language.

While the Examiner is correct in noting that "Java" is a registered trademark, Applicant is not using "Java" as a claim limitation. Rather, Applicant recites a "JAVASERVER page." JAVASERVER Pages (JSP) is a Java technology that allows software developers to dynamically generate HTML, XML or other types of documents in response to a Web client request. The technology allows Java code and certain pre-defined actions to be embedded into static content. In contrast to the concerns expressed in the Examiner's cited case *In re Simpson*, "JAVASERVER page" identifies the goods themselves, and not merely the source of goods, i.e., JAVA® or JAVASERVER® assigned to Sun Microsystems. Furthermore, "JAVASERVER page" is not a registered trademark.

In light of the comments presented herewith, the rejection has been traversed. "Javaserver page" is not indefinite, and does not raise the same concerns, that "JAVA" does. Withdrawal of the rejection is therefore requested.

II. 35 U.S.C. § 101

The Examiner rejected claims 12-17, 19-22, and 23-33 under 35 U.S.C. § 101 as directed towards non-statutory subject matter. Applicant amended claims 12 and 23 accordingly, thereby overcoming the Examiner's rejection. Support for the amendments can be found on pages 26 of the as-filed specification.

III. 35 U.S.C. § 102, Anticipation

The Examiner rejected claims 1-34 under 35 U.S.C. § 102 as anticipated by *Murren et al.*, Method and System for Customization of a Program, U.S. Patent 7,000,185 (February 14, 2006) (hereinafter *Murren*). This rejection is respectfully traversed. With regard to claim 1, the Examiner states:

> As per claim 1, Murren discloses method in a data processing system for processing a Java server page (e.g. FIG. 1A and related text), the method comprising:
>
> translating the Java server page into a document object model object (e.g. FIG. 1 A, SOURCE JSP (101) to DOM (105) and related text);
>
> configuring a set of visitor classes for invocation in a selected sequence (col. 11 :55-65 ". . . component instantiates a tag object for . . . object class . . ." and e.g. FIG. 7, 701 -702 and related text); and
>
> processing the document object model using the set of visitor classes in the selected sequence to perform a desired set of custom functions on the document object model (col. 4:25- 35 ". . . traverses the DOM . . ." and col. 9: 10-25 ". . . passing the root of the node of the DOM version of the source JSP . . . ").

Office Action of March 20, 2007, p.5.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). In this case, each and every feature of the presently claimed invention is not identically shown in the cited reference, arranged as they are in the claims.

Claim 1 has been amended to include the features of original, now canceled, claim 7. As amended, claim 1 is as follows:

> 1. A method in a data processing system for processing a Javaserver page, the method comprising:
>
> translating the Javaserver page into a document object model object;
>
> configuring a set of visitor classes for invocation in a selected sequence;

processing the document object model using the set of visitor classes in the selected sequence to perform a desired set of custom functions on the document object model; and

storing results, as processing the document object model object occurs by selected method in the methods, in a hash map, wherein the results in the hash map are used by subsequently invoked methods, wherein the subsequently invoked method does not convert the results into a second document object model object.

Murren does not teach all of the features of claim 1 as amended, because *Murren* does not teach "storing results, as processing the document object model object occurs by selected method in the methods, in a hash map, wherein the results in the hash map are used by subsequently invoked methods, wherein the subsequently invoked method does not convert the results into a second document object model object." Support for the claim amendments can be found in original, now canceled, claim 7, and on page 21, lines 22-28 of the as-filed specification.

Murren teaches a method of customizing content for a web application. A DOM version of the JSP web application can be converted into a variety of different languages by using a customization bundle. When the content is executed, the custom content is retrieved based on a user's specified preferences. Specifically, *Murren* teaches:

> The customization system "compiles" a computer program (e.g., JSP or ASP) by identifying the content of each statement (e.g., tag) of the computer program that may be customized. The customization system identifies the type (e.g., JSP tag) associated with each statement. Based on the type of statement, the customization system identifies content of the statement that can be customized and stores the identified content in a custom content bundle. A custom content bundle (e.g., resource bundles provided for in the Java platform released by Sun Microsystems) contains name and value pairs that map content identifiers to the corresponding content. The customization system then replaces the identified content in the statement with an include content command (e.g., an include content tag when the program is a JSP) that includes the content identifier of the corresponding content. When the computer program is executed, the include content command of the statement causes the retrieving of a content associated with its content identifier from a custom content bundle. Additional custom content bundles may be created to support the customization. For example, the initial custom content bundle may support English, and additional custom content bundles may support French, German, and so on. When the computer program is executed, the appropriate custom content bundle is accessed to effect the customization of the computer program. In this way, computer programs can be automatically "compiled" to support the customization, such as localization.

Murren, col. 2, l. 55-col. 3, ll. 15.

In support of the present rejection, the Examiner cites the following passages:

FIG. 1B is a block diagram illustrating the compilation and execution of a JSP by the customization system in one embodiment. The customization compiler 106 compiles a DOM version 105 of the source JSP into a target JSP 109. The customization compiler also generates a custom content bundle 110. The customization compiler inputs HTML tag instructions 107 and custom tag instructions 108. The tag instructions specify how to customize each type of tag of a JSP. The customization compiler traverses the DOM tree compiling each tag.

...

FIGS. 2 5 are flow diagrams illustrating the processing of the customization compiler in one embodiment. FIG. 2 is a flow diagram illustrating the processing of a compile tag component in one embodiment. The compile tag component is invoked by the customization compiler passing the root node of the DOM version of the source JSP. The component compiles that tag and recursively invokes the compile tag component for each child tag. In decision block 201, if the passed node represents an HTML tag, then the component continues at block 202, else the component continues at block 203. In block 202, the component invokes the compile HTML tag component and continues at block 205. In decision block 203, if the passed node represents a JSP tag, then the component continues at block 204, else the component continues at block 205.

...

FIG. 5 is a flow diagram illustrating the processing of the generate replacement component in one embodiment. This component replaces content of a tag with an include content tag and stores the content in the custom content bundle. In block 501, the component creates a new content identifier. In one embodiment the content identifier is the name of the DOM version of the JSP followed by a unique number (e.g., user-search-1, user-search-2, and so on). In block 502, the component retrieves the content from the attribute or the PCDATA in accordance with the tag instructions. In block 503, the component adds a name and value pair to the custom content bundle that maps of the content identifier to the retrieved content. In block 504, the component generates the include content tag replacement for the content. In block 505, the component stores the replacement for the content into the tag and then returns.

...

In decision block 606, if all the child tags of the passed tag have already been selected, then the component returns, else the component continues at block 607. In block 607, the component recursively invokes the execute tag component passing the selected child tag and then loops to block 605 to select the next child tag.

FIG. 7 is a flow diagram illustrating the processing of the execute JSP tag component in one embodiment. The component is passed a JSP tag and instantiates an object associated with the passed tag, sets the attributes of the object, and invokes a method of the object to generate the HTML associated with the passed tag.

...

block 705, the component invokes the appropriate set attribute method of the tag object to set the attribute value specified in the tag. The selected attribute may have an include content tag. If so, before setting the attribute

> value, the component retrieves the corresponding content from the custom content bundle and sets the attribute with the retrieved content. The component then loops to block 703 to select the next attribute of the tag. In block 706, the component invokes the processing method of the tag object (e.g., doStartTag) to allow the tag object to generate of the HTML corresponding to the tag. The component then returns.

Murren, col 4, ll. 25-35; col. 9, ll. 10-25; col. 11, ll. 9-2; col. 12, ll. 50-58; and col. 12, ll. 8-19.

Nowhere in cited portions, or elsewhere, does *Murren* disclose the features of amended claim 1. That is, *Murren* does not disclose "storing results, as processing the document object model object occurs by selected method in the methods, in a hash map, wherein the results in the hash map are used by subsequently invoked methods, wherein the subsequently invoked method does not convert the results into a second document object model object."

With regard to the Examiner's statement regarding original, now canceled, claim 7, the cited portions of *Murren* discloses only that the specific tags for customization modules are updated to reflect amended contents of the module (FIG 5). The cited portions further teach that type and application name pairs are linked in a table to the corresponding base name for the custom content bundles (col. 12, ll. 47-58). The cited portions finally teach that attributes of a tag can be processed sequentially (col. 12, ll. 8-16). Nothing in the cited passages teach the features now incorporated into claim 1, namely "storing results, as processing the document object model object occurs by selected method in the methods, in a hash map, wherein the results in the hash map are used by subsequently invoked methods, wherein the subsequently invoked method does not convert the results into a second document object model object."

Because *Murren* does not teach each feature of claim 1, *Murren* does not anticipate claim 1 as amended. In light of the amendments to claim 1, and the comments presented herein, the rejection to claim 1 has been overcome.

Claims 2-11 depend from claim 1 and therefore incorporate all the features of claim 1. By virtue of their dependence from claim 1, the combination of *Murren* does not anticipate claims 2-11. Therefore, the rejection of claims 2-11 under 35 U.S.C. § 102 has been overcome.

Claims 12, 23 and 34 are drawn to an embodiment having features consistent with the features of claim 1, and have been amended to be consistent with the amendments to claim 1. *Murren* does not disclose each feature of claims 12, 23 or 34. Therefore, *Murren* does not anticipate claims 12, 23, and 34 as amended.

Claims 13-22 and 24-33 depend from claims 12 and 23 respectively. Claims 13-22 and 24-33 therefore incorporate all the features of their underlying respective independent claims. By virtue of their dependence from claims 12 and 23 respectively, *Murren* does not anticipate claims

13-22 and 24-33. Therefore, the rejection of claims 13-22 and 24-33 under 35 U.S.C. § 102 has been overcome.

IV. Conclusion

The subject application is patentable over *Murren* and should now be in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: June 20, 2007

Respectfully submitted,

/Theodore D. Fay, III/

Theodore D. Fay, III
Reg. No. 48,504
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

TF/bw